

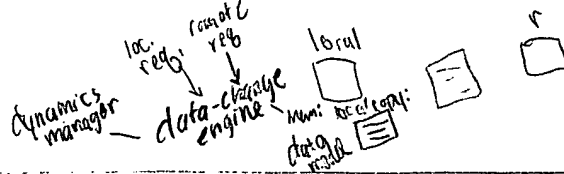
out of order  
roll back  
redo/remake  
undo/unmake  
save undone  
insert make

## Amendments to the Claims

7/19/99

This listing of claims will replace all prior versions, and listings, of claims in the application:

### Listing of Claims:



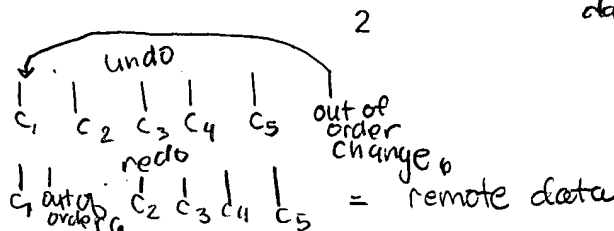
1. (Presently Amended) A local network-capable device adapted for collaborative operation and communication over a network with at least one remote network-capable device to enable the local network-capable device and the remote network-capable device to cooperatively edit the same data, said local network-capable device comprising:

- A) a memory for storing a local copy of the data in accordance with a data model;
- B) a data-change engine coupled with the memory, and responsive to a plurality of data change requests, for controlling storage of the local copy of data in the memory in accordance with the data model and making changes to the local copy of the data; the data change requests including a locally-generated data change request and a remotely-generated data change request; and

- C) a dynamics manager, coupled with the data-change engine, and responsive to the data change requests for controlling the engine and coordinating execution of the data change requests; wherein the dynamics manager, responsive to the data change requests, can cause the making of selected data changes in an order, and, responsive to a data change request being received out of the order, the rolling-back undoing of the selected data changes to a point where a data change corresponding to the out-of-order data change request should have been made and the remaking of the selected undone data changes in another order so that the local copy of the data is consistent with a copy of the data maintained by the remote network-capable device.

processed.

confusing - there is no mention of data being maintained at remote device.



2

make → undo → remake

1 2. (Presently Amended) The local network-capable device in accordance with claim  
2 1, wherein the dynamics manager causes making, ~~rolling back~~ undoing and  
3 remaking of data changes in response to a data change request priority scheme.

1 3. (Original) The local network-capable device in accordance with claim 1, wherein  
2 the data change request priority scheme includes encoding the data change  
3 requests with request sequence numbers, and the dynamics manager is  
4 responsive to the request sequence numbers in determining an order for making  
5 data changes specified by the data change requests.

1 4. (Original) The local network-capable device in accordance with claim 3, wherein  
2 the data change request priority scheme includes encoding the data change  
3 requests with an identifier corresponding to a characteristic of the network-  
4 capable device that generated the request, and the dynamics manager is  
5 responsive to the identifier in causing making of data changes.

1 5. (Original) The local network-capable device in accordance with claim 4, wherein  
2 each network capable device and a user thereof corresponds to an endpoint, and  
3 the identifier comprises an endpoint number corresponding to the endpoint that  
4 originated the data change request.

1 6. (Original) The local network-capable device in accordance with claim 5, wherein  
2 the request sequence numbers comprise endpoint relative sequence numbers,  
3 and the dynamics manager causes the data change requests to be processed in  
4 an order dependent on the endpoint relative sequence numbers and the endpoint  
5 numbers.

1 7. (Presently amended) The local network-capable device in accordance with claim  
2 3, wherein the data change request priority scheme includes a dependency

3 identifier in each data change request, and the dynamics manager is responsive  
4 to the dependency identifier in causing ~~rolling-back~~ undoing and remaking of data  
5 changes.

1 8. (Previously amended) The local network-capable device in accordance with  
2 claim 7, wherein the dependency identifier specifies at least one data change  
3 request on which the encoded data change request depends.

1 9. (Original) The local network-capable device in accordance with claim 8, wherein  
2 the dependency identifier specifies one data change request on which encoded  
3 data change request depends.

1 10. (Original) The local network-capable device in accordance with claim 8, wherein  
2 the dynamics manager executes do, undo and redo operations with respect to  
3 data change requests to ensure that each one of the data change requests is  
4 processed only after the specified data change request on which the one data  
5 change request depends has been processed.

1 11. (Presently Amended) The local network-capable device in accordance with claim  
2 8, wherein  
3 A) the request sequence numbers comprise endpoint relative sequence  
4 numbers;  
5 B) the dynamics manager causes the data change requests to be processed  
6 in an order dependent on the endpoint relative sequence numbers and the  
7 endpoint numbers; and  
8 C) the dynamics manager causes data changes to be ~~rolled-back~~ undone  
9 and remade responsive to the dependency identifier.

1 12. (Presently Amended) A distributed, coordinated system for maintaining plural  
2 copies of the same data pursuant to a distributed data model, wherein the copies  
3 can be changed responsive to users' actions, the system comprising:

4 A) a plurality of computer systems; each of the computer systems capable of  
5 locally generating a plurality of data change requests for changing a local  
6 copy of the data and of executing data change requests including the  
7 locally-generated data change requests and remotely-generated data  
8 change requests generated by others of the computer systems so as to  
9 make the requested changes to the local copy of the data;

10 B) each of the computer systems including a dynamics manager that  
11 receives locally-generated data change requests and remotely-generated  
12 data change requests for determining, responsive to information contained  
13 in the locally-generated and remotely-generated data change requests, an  
14 order in which requested changes are made to the local copy of the data,  
15 for making requested data changes in the determined order, for rolling-  
16 back undoing data changes made when a remotely-generated data  
17 change request is received out of the determined order to the point where  
18 a data change corresponding to the out-of-order data change request can  
19 be made in the determined order and for remaking requested-undone data  
20 changes in a new order.

1 13. (Original) The system in accordance with claim 12, wherein the dynamics  
2 manager of each computer system is responsive to data dependency information  
3 and request sequence information recorded in the data change requests in  
4 determining the order in which the requested changes are made to the local copy  
5 of the data; the data dependency information comprising an indication of at least  
6 one prior data change request on which the data change request depends; and  
7 the request sequence information indicating a sequential position of the data  
8 change request among a plurality of data change requests generated by the  
9 computer system that generated the data change request.

14.

(Presently Amended) A framework apparatus for providing communication services for an activity-based collaboration system in which data change requests comprising deltas are communicated over a network between network-capable devices in order to maintain consistency between local data copies, the framework apparatus comprising a communications manager operable on a local network capable device for sending locally-generated deltas over a network to at least one remote network-capable devices and for receiving remotely-generated deltas from the at least one remote network-capable device; and a dynamics manager responsive to dependency information contained in the deltas for determining an order for processing the deltas to make changes to each local data copy, the dynamics manager being responsive to the reception of a remotely-generated delta out of the determined order for causing the selective rollback undoing of changes made to a local data copy, the making of a data change corresponding to the out-of-order delta and the selective remaking of the rolled-back undone data changes in a new order.

15.

(Presently Amended) A method for providing communication services for an activity-based collaboration system, in which data change requests comprising deltas are communicated over a network between network-capable devices in order to maintain consistency between local data copies, the method comprising the steps of:

- A) sending locally-generated deltas from a local network-capable device over a network to at least one remote network-capable devices and for receiving remotely-generated deltas from the at least one remote network-capable device;
- B) determining an order for processing the deltas based on sequence information contained within the deltas;
- C) processing the deltas in the determined order thereby making changes to a local data copy as requested by the deltas;

- 14 D) ~~rolling-back~~ undoing requested changes made to the local data copy in  
15 response to sequence information contained within the deltas indicating  
16 that a remotely-generated delta has been received out of the determined  
17 order; and  
18 E) making a data change to the local data copy corresponding to the out-of-  
19 order delta; and  
20 F) remaking the rolled-back undone data changes in a new order.

Claim 16 (Canceled).

Claims 17-34 (Withdrawn).

- 1 35. (New) A method for processing data change requests in a network-capable  
2 device having a local data copy and data change means for generating data  
3 change requests, each data change request specifying a change to be made to  
4 the local data copy, wherein the network-capable device communicates over a  
5 network with at least two other network-capable devices to enable all network-  
6 capable devices to cooperatively edit the same data during an editing session,  
7 the method comprising:  
8 (a) when a first and second received data change requests require that a third  
9 data change have previously been made to the local data copy,  
10 determining an order of making the data changes as specified by the first  
11 and second data change requests based on information contained in the  
12 data change requests;  
13 (b) when a change specified by one of the first and second data change  
14 requests cannot be made in the order determined by the data change  
15 requests because other changes have already been made to the local  
16 data copy, undoing the other changes to the local data copy;  
17 (c) making a change to the local data copy as specified by the one data  
18 change request; and

how can  
changes  
be made  
w/out the  
third

19 (d) redoing data changes undone in step (b) so that data changes are made  
20 to the local data copy consistently with changes made to local data copies  
21 in all network-capable devices.

1 36. (New) The method of claim 35 wherein step (a) comprises:

2 (a1) when a first and second received data change requests require that a third  
3 data change have previously been made to the local data copy and the  
4 third change has not been made, placing the first and second received  
5 data change requests in a holding queue for subsequent processing.

1 37. (New) The method of claim 36 wherein step (b) comprises:

2 (b1) when the changes specified by the first and second data change requests  
3 can be made in the order determined by the data change requests making  
4 the data changes specified by the data change requests in the determined  
5 order.

1 38. (New) The method of claim 37 further comprising:

2 (e) determining whether changes to the local data copy specified by data  
3 change requests in the holding queue can be made.

1 39. (New) The method of claim 38 wherein the information in the data change  
2 requests comprises an order in which each of the network-capable devices  
3 joined the editing session.

1 40. (New) The method of claim 35 wherein the information in the data change  
2 requests further comprises a sequence number for each network-capable device.

1 41. (New) The method of claim 35 further comprising:

2 (e) placing information for each change made to the local data copy in a data  
3 change log; and

4 (f) using the information in the data change log to undo changes to the local  
5 data copy.

1 42. (New) The method of claim 41 further comprising:

2 (g) after each data change has been made, updating a timestamp vector  
3 comprising a data change sequence number for each endpoint.

1 43. (New) The method of claim 42 wherein step (a) comprises:

2 (a2) when a first and second received data change requests require that a third data  
3 change have previously been made to the local data copy, using the timestamp  
4 vector to determine whether the third change has been made.

1 44. (New) Apparatus for processing data change requests in a network-capable  
2 device having a local data copy and data change means for generating data  
3 change requests, each data change request specifying a change to be made to  
4 the local data copy, wherein the network-capable device communicates over a  
5 network with at least two other network-capable devices to enable all network-  
6 capable devices to cooperatively edit the same data during an editing session,  
7 the apparatus comprising:

8 when a first and second received data change requests require that a third  
9 data change have previously been made to the local data copy, means for  
10 determining an order of making the data changes as specified by the first and  
11 second data change requests based on information contained in the data change  
12 requests;

13 when a change specified by one of the first and second data change  
14 requests cannot be made in the order determined by the data change requests  
15 because other changes have already been made to the local data copy, means  
16 for undoing the other changes to the local data copy;

17 means for making a change to the local data copy as specified by the one  
18 data change request; and



19 means for redoing undone data changes so that data changes are made  
20 to the local data copy consistently with changes made to local data copies in all  
21 network-capable devices.

1 <sup>36</sup> 45. (New) The apparatus of claim 44 wherein the means for determining an order of  
2 making the data changes comprises:  
3 a holding queue; and  
4 means operable when a first and second received data change requests  
5 require that a third data change have previously been made to the local data  
6 copy and the third change has not been made, for placing the first and second  
7 received data change requests in the holding queue for subsequent processing.

1 <sup>37</sup> 46. (New) The apparatus of claim 45 further comprising means operable when the  
2 changes specified by the first and second data change requests can be made in  
3 the order determined by the data change requests for making the data changes  
4 specified by the data change requests in the determined order.

1 <sup>38</sup> 47. (New) The apparatus of claim 46 further comprising means for determining  
2 whether changes to the local data copy specified by data change requests in the  
3 holding queue can be made.

1 <sup>39</sup> 48. (New) The apparatus of claim 47 wherein the information in the data change  
2 requests comprises an order in which each of the network-capable devices  
3 joined the editing session.

1 <sup>40</sup> 49. (New) The apparatus of claim 44 wherein the information in the data change  
2 requests further comprises a sequence number for each network-capable device.

1 <sup>41</sup> 50. (New) The apparatus of claim 44 further comprising:  
2 means for placing information for each change made to the local data  
3 copy in a data change log; and

4 mean for using the information in the data change log to undo changes to  
5 the local data copy.

1 <sup>42</sup> 51. (New) The apparatus of claim 50 further comprising means operable after each  
2 data change has been made for updating a timestamp vector comprising a data  
3 change sequence number for each endpoint.

1 <sup>43</sup> 52. (New) The apparatus of claim 51 wherein the means for determining an order of  
2 making the data changes comprises means operable when a first and second  
3 received data change requests require that a third data change have previously  
4 been made to the local data copy for using the timestamp vector to determine  
5 whether the third change has been made.

---

### Remarks and Arguments

Claims 1-15 have been presented for examination. Claims 1, 2, 7, 11, 12, 14 and 15 have been amended. New claims 35-52 have been added.

Claims 1-4, 12, 14 and 15 have been rejected as obvious under 35 U.S.C. §103(a) over U.S. Patent No. 5,806,075 (Jain) in view of U.S. Patent No. 6,449,622 (LaRue). The examiner comments that Jain discloses the claimed subject matter with the exception that Jain does not show receiving data change requests out of order. [The examiner asserts that mechanisms for dealing with the reception of out-of-order data change requests is well-known in the art as evidenced by LaRue.] Although LaRue relates to synchronizing datasets rather than peer-to-peer collaboration systems, the examiner claims that the LaRue system relates to analogous art and discloses the coordination of data change requests that are received out of order. The examiner claims that a person skilled in the art would have recognized that incorporation of the features of LaRue into the Jain system would allow Jain to operate in high latency environments.

As previously discussed, in the present invention, a dynamics manager in each collaborator's computer implements a data change request priority scheme that determines an order for executing the data change requests to promote data consistency. If a data change request from a remote collaborator arrives at a local collaborator, the dynamics manager can determine (from the order and dependency information in the new request) that this newly-received data change request should have been made before other already-made data change requests. In this case, the dynamics manager effectively reorders the data change requests and makes the changes to the local data copy in the correct order. It does this by selectively undoing data changes that have already been made by "undoing" the effects of these changes change-by-change until the local data copy is in a state at which the newly received data change request should have been made. Each individual change can be undone because the dynamics manager saves the change and information to undo the change in a delta log.

make -undo-remake → insert an action

After undoing the changes, the dynamics manager causes the newly received data change to be made and then remakes all of the subsequent data changes thereby resulting in a new order of data changes that incorporates the newly-received change. The result is that conflicts are resolved and the data changes are made in the correct order without user intervention.

The Jain reference discloses a procedure level replication technique that applies remote procedure calls (RPCs) to modify a replicated data copy and can address conflicts that occur during the processing of the RPCs and "rollback" any modifications made when a conflict is detected. In Jain, the processing of RPCs is "atomic", that is, the modification performed by the RPC is either made permanent or not performed. Jain discloses two methods for processing RPCs: transactional processing and non-transactional processing. Transactional processing processes a series of RPCs whereas non-transactional processing processes only a single RPC.

In both types of processing, a "savepoint" is established that saves the state of the data before one or more RPCs are applied. The savepoint is established for transactional processing as set forth in column 20, lines 7-13 and the savepoint for non-transactional processing is discussed at column 21, line 46. In transaction processing, if all RPC(s) complete without exception, then the transaction is "committed" and the changes made permanent as set forth at Jain column 20, lines 25-27 and at column 21, lines 58-60. Alternatively, if an exception occurs, then all of the transaction are rolled back to the savepoint as set forth in Jain, column 21, lines 11-13 and column 21, lines 49-50, and an entry is made in the exceptions table. After an exception occurs due to an update conflict, error handling can then be initiated. This error handling is described at Jain column 22, lines 47-55, which states that "The subsequent error processing can be done with various degrees of operator intervention and automation."

after exc.  
error  
handling  
does to  
redo the  
undone changes

Jain discloses no mechanism for handling an RPC that might arrive "out-of-order" after RPCs have been committed because changes have been made permanent at that point. [In particular, Jain discloses no mechanism for "undoing" changes on a change-by-change basis since Jain does not store undo information for each change and only stores the data state at the last savepoint.]

col. 22

col. 22

no undoing ✓ yes, undoing - col. 22

Jain - no remaking - col. 22  
redo log.

state corresponds to changes  
corresponding to out of order data change

no out of order - col. 22

Jain does show undoing changes - just b/c they are not made permanent does not mean they have not been made in the database. although they are not committed, they are made, the commit cycle allows for a redo.

The LaRue reference supplements the Jain disclosure by disclosing that conflicts can be automatically handled by comparing timestamps on the conflicting changes when both are present during processing and implementing the change with the latest timestamp value, a "latest change wins" approach described at LaRue, column 47, lines 13-19. However, if this approach does not succeed because the timestamps cannot be compared, then LaRue also uses user intervention as described at LaRue, column 47, lines 20-28. LaRue also does not address the situation where a new "out-of-order" change arrives after all other changes have been committed. Specifically, LaRue discloses no mechanism for "undoing" changes on a change-by-change basis since LaRue does not store undo information for each change and only stores the data state at the last savepoint. no making  
mon undoing

Consequently, the combination of Jain and LaRue might suggest using a "latest change wins" approach to handling conflicts are an alternative to the rollback disclosed in Jain, but cannot teach or suggest undoing (each) change on a change-by-change basis since neither reference teaches or suggests this.

In order to prevent confusion regarding the "rollback" disclosed in Jain with the change-by-change undoing of the present invention, the claims have been amended to replace the term "rollback" with undo. For example, amended claim 1 recites, in lines 17-22, "...the making of selected data changes in an order, and, responsive to a data change request being received out of the order, the undoing of the selected data changes to a point where a data change corresponding to the out-of-order data change request should have been made and the remaking of the undone data changes in another order so that the local copy of the data is consistent with a copy of the data maintained by the remote network-capable device." Therefore, amended claim 1 clearly recites that data changes are undone back to the point where an out-of-order change should have been made and then the changes are redone to make the changes in a new order. As discussed above, the combination of Jain and LaRue cannot perform such an operation and does not suggest such an operation. Thus, amended claim 1 clearly distinguishes over the combination of the cited references.

Similar changes have been made in independent claims 12, 14 and 15 and these claims distinguish over the cited references in the same manner as amended claim 1.

Amendments have been made to claims 2, 7 and 11 to conform them to the changes made in claim 1.

Claims 2-4 are dependent, either directly or indirectly, on amended claim 1 and incorporate the limitations thereof. Therefore, they distinguish over the cited combination in the same manner as amended claim 1. In addition, these claims recite additional limitations not shown or suggested by the cited combination. For example, claim 2 recites the undo and remaking operations recited in claim 1 and not taught or suggested by the cited combination.

Claims 5-11 and 13 have been rejected under 35 U.S.C. §103(a) as obvious over the Jain and LaRue references and further in view of U.S. Patent No. 5,802,322 (Niblett.) The examiner comments that Jain discloses most of the claimed material except that it does not disclose the use of endpoint numbers as identifiers. However, the examiner claims that Niblett shows such endpoint numbers and their use as identifiers. The examiner concludes that it would have been obvious to combine Jain, LaRue and Niblett in order to more completely serialize updates.

As mentioned in the response to a previous office communication, the Niblett patent discloses a token passing conference ring network in which a "permit-token" is used to serialize updates. When updates are performed, the permit-token is used to establish an order for the updates. In this manner, each node in the system will know when an update is missing. Consequently, the node can wait until the missing update has been received before applying later updates. This operation is clearly set forth in the Niblett abstract.

Claims 5-11 depend on amended claim 1 and incorporate the limitations thereof. The combination of Jain, LaRue and Niblett references does not suggest the make, undo and remake operations that are recited in amended claim 1. As discussed above, Jain and LaRue do not teach or suggest the recited combination. The addition of Niblett does not change this conclusion. This is because the Niblett serialization mechanism guarantees that an update cannot arrive "out-of-order" after other updates have been applied as in the present invention. Therefore, there is no reason to undo updates and then redo them in a different order as recited in amended claim 1 lines 17-22 as discussed above and Niblett does not mention any such operation. Consequently, the

combination of Jain, LaRue and Niblett cannot teach or suggest the recited limitations since no of the reference teaches or suggests the recited combination. Therefore, claims 5-11 patentably distinguishes over the cited combination of Jain, LaRue and Niblett.

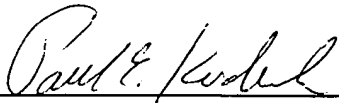
Claim 13 is dependent on amended claim 12 and incorporates the limitations thereof. Since amended claim 12 recites essentially the same limitations as amended claim 1, claim 13 distinguishes over the cited combination in the same manner as amended claim 1.

New claims 35-52 have been added to more specifically recite the conditions under which the make, undo and redo operations are processed. These claims are based on Figures 11A, 11B and 12 and the accompanying description at page 33, line 14 to page 38, line 17. The two independent claims 35 and 44 have similar scope and claim 35 is representative of that scope. Claim 35 recites, "...making the data changes as specified by the first and second data change requests..." (lines 10-11); "when a change specified by one of the first and second data change requests cannot be made in the order determined by the data change requests because other changes have already been made to the local data copy, undoing the other changes to the local data copy..."(lines 13-16); making a change to the local data copy as specified by the one data change request; (lines 17-18) and "...redoing data changes undone ...so that data changes are made to the local data copy consistently with changes made to local data copies in all network-capable devices."(lines 19-21). Consequently, claims 35 and 44 recite the make, undo and redo operations that, as discussed above, are not taught or recited in Jain, LaRue and Niblett. Consequently, these claims patentably distinguish over the cited references.

Claims 36-43 are dependent, either directly or indirectly, on new claim 35 and incorporate the limitations thereof. Therefore, they distinguish over the cited combination in the same manner as new claim 35. Similarly, claims 45-52 are dependent, either directly or indirectly, on new claim 44 and incorporate the limitations thereof. Therefore, they distinguish over the cited combination in the same manner as new claim 44.

In light of the forgoing amendments and remarks, this application is now believed in condition for allowance and a notice of allowance is earnestly solicited. If the examiner has any further questions regarding this amendment, he is invited to call applicants' attorney at the number listed below. The examiner is hereby authorized to charge any fees or direct any payment under 37 C.F.R. §§1.17, 1.16 to Deposit Account number 02-3038.

Respectfully submitted



Date: \_\_\_\_\_

11/26/03

Paul E. Kudirka, Esq. Reg. No. 26,931  
KUDIRKA & JOBSE, LLP  
Customer Number 021127  
Tel: (617) 367-4600 Fax: (617) 367-4656